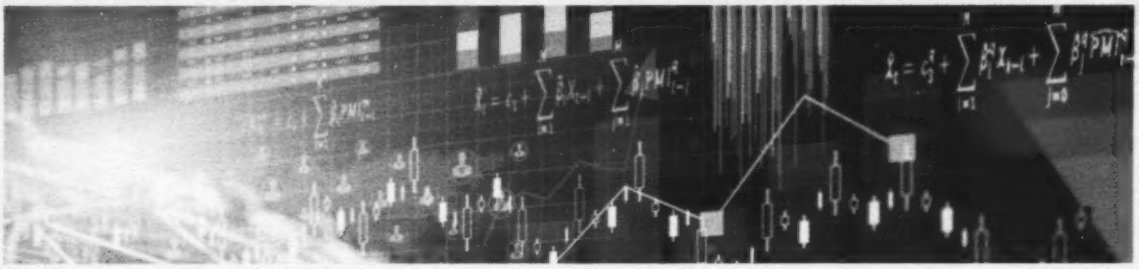
BANK OF CANADA
BANQUE DU CANADA

Staff Working Paper/Document de travail du personnel 2018-34

Incentive Compatibility on the Blockchain



by Jonathan Chiu and Thorsten V. Koeppl

Bank of Canada staff working papers provide a forum for staff to publish work-in-progress research independently from the Bank's Governing Council. This research may support or challenge prevailing policy orthodoxy. Therefore, the views expressed in this paper are solely those of the authors and may differ from official Bank of Canada views. No responsibility for them should be attributed to the Bank.

www.bank-banque-canada.ca

Bank of Canada Staff Working Paper 2018-34

July 2018

Incentive Compatibility on the Blockchain

by

Jonathan Chiu[1] and Thorsten V. Koeppl[2]

[1] Funds Management and Banking Department
Bank of Canada
Ottawa, Ontario, Canada K1A 0G9
jchiu@bankofcanada.ca

[2] Department of Economics
Queen's University
Kingston, Ontario, Canada K7L 3N6
thor@econ.queensu.ca

ISSN 1701-9397

© 2018 Bank of Canada

Acknowledgements

We thank our discussant, Larry Glosten, the audience of the RFS FinTech conference, two anonymous referees and the editor for their comments. This research was supported by SSHRC Insight Grant 435-2014-1416. The authors declare that they have no relevant or material financial interests that relate to the research described in this paper.

Abstract

A blockchain is a digital ledger that keeps track of a record of ownership without the need for a designated party to update and enforce changes to the record. The updating of the ledger is done directly by the users of the blockchain and is traditionally governed by a proof-of-work (PoW) protocol. We formalize this protocol as a Cournot game where users compete to update the blockchain for a reward. Cheating occurs in the form of "double spending" when users try to tamper with ownership records in order to defraud their counterparties. Ruling out incentives to cheat can be summarized in the form of a "no double-spending constraint." These constraints put restrictions on the design of a blockchain and, thus, play a role akin to incentive compatibility constraints in classic mechanism design.

Bank topics: Digital currencies; Payment clearing and settlement systems; Economic models
JEL codes: G2, H4, P43

Résumé

Une chaîne de blocs est un grand registre numérique qui suit l'évolution des opérations faites par les auteurs de transactions sans devoir faire appel à un tiers chargé de mettre à jour et à exécution les changements consignés. La mise à jour du registre est faite directement par les utilisateurs de la chaîne de blocs et est habituellement régie par un protocole, la preuve de travail. Nous formalisons ce protocole comme un jeu de Cournot où les utilisateurs s'affrontent pour mettre à jour la chaîne en échange d'une récompense. Il y a tricherie – sous la forme d'une « double dépense » – lorsque les utilisateurs tentent de falsifier les transactions consignées afin d'escroquer leurs contreparties. Il est possible d'écarter ces incitations à tricher en empêchant l'inscription en double des dépenses. Cette entrave soumet à des restrictions la conception d'une chaîne de blocs et, par conséquent, joue un rôle analogue à celui des contraintes de compatibilité incitative auxquelles est assujettie la conception des mécanismes classiques.

Sujets : Monnaies numériques; Systèmes de compensation et de règlement des paiements; Modèles économiques
Codes JEL : G2, H4, P43

Non-Technical Summary

A blockchain is a decentralized ledger that digitally records the ownership and transfer of assets. Owing to its digital nature, an ownership record is merely a string of bits that can easily be copied and reused repeatedly, leading to a double-spending problem.

A key challenge for designing the blockchain system is to discourage double-spending attempts so that users trust the information contained in the ledger. The security of a blockchain is based on three elements: (i) a consensus protocol, (ii) confirmation lags, and (iii) a reward scheme.

First, to reach a common consensus about the new update among all users in a decentralized setting, validators are asked to compete for the right to append a new block to the chain. While this competition can take various forms, the most common consensus protocol is proof-of-work (PoW). This competition process is called mining.

Second, the final settlement of a transaction is often intentionally delayed by introducing a confirmation lag, which increases the difficulty of completing a double-spending attempt and hence prevents users from altering the history of transactions.

Third, as the mining process is costly, validators need to be provided the proper incentive. A blockchain system often offers rewards financed by seigniorage – issuing cryptocurrency or tokens – or by collecting transaction fees from traders.

To capture these key elements, we model the PoW protocol as a simple Cournot game of mining. We then formalize the double-spending problem and show that it can be summarized as a simple incentive compatibility constraint. We briefly describe how this constraint manifests itself in two examples.

The first example concerns a securities settlement system where both the asset transfer and the payment are recorded on a blockchain (Chiu and Koeppl, 2018). In such a system, it is important to avoid settlement failures where the seller of a security fails to deliver the security while receiving payment, or the buyer of a security fails to deliver payment while receiving the security. A delivery versus payment (DvP) mechanism typically ensures that the security and the cash are exchanged simultaneously in order to avoid such a settlement failure.

In the second example, we consider a blockchain used to record cryptocurrency transfers when purchasing real goods (Chiu and Koeppl, 2017). DvP in this context is not automatic anymore as the ownership of goods is not recorded digitally on the blockchain. Since there is no DvP, a seller can make double spending more difficult by introducing a confirmation lag. The goods are to be delivered only after the transaction has been confirmed sufficiently many times in the blockchain.

1 Blockchain as a Distributed Ledger

A blockchain is a decentralized ledger that digitally records the ownership and transfer of assets. Owing to its digital nature, an ownership record is simply a string of bits that can easily be copied and reused repeatedly, leading to a *double-spending problem*. In a centralized system such as PayPal, this problem can be solved by relying on a trusted third party to manage the ledger. This trusted central authority validates and enforces all transactions, preventing users from tampering with the ledger.

Blockchain systems aim to maintain a digital ledger without the need for a designated party to keep records and enforce the transfer of ownership. Instead, transactions are verified and processed by a network of potentially anonymous validators. In this system, a *block* is simply a set of transactions that transfer ownership between users in the ledger. From individual blocks a chain is formed by *time-stamping*. The blocks are linked together in a sequence where each block depends on the previous block in time. This creates a full, historical record of transactions where no past block can be changed without also changing all the subsequent blocks.

The ownership of assets is protected by the use of basic cryptographic principles. An owner holds a private-public key pair where the private key is kept secret and controls the entry in the blockchain, while the public key proves ownership to any other party. A transaction is conducted by transferring an entry in the blockchain to a different private-public key pair.

In a truly decentralized blockchain such as Bitcoin, anyone can access the public ledger, verify its information and even serve as a validator who updates the chain with a new block. The blockchain itself lives on a distributed network where users interact peer-to-peer and each keeps a copy of the ledger. This extreme redundancy introduces resiliency into the system. As long as one of the peers is live on the network the ledger can be accessed and transactions can be conducted.

The key challenge is to design the rules for updating the blockchain so that it is hard to tamper with and, thus, users trust the information contained in the ledger. The security of a blockchain is based on three elements: (i) a consensus protocol, (ii) confirmation lags, and (iii) a reward scheme.

First, in a decentralized network, the system needs to ensure that when new transactions are incorporated into the blockchain, a common consensus about the new update is agreed upon among all users. To reach such a consensus in a decentralized setting, validators are asked to compete for the right to append a new block to the chain. This competition can take various forms. In the

most common consensus protocol, proof-of-work (PoW), this process is called mining. Validators, also called *miners*, need to solve a computationally difficult problem. The winner of this mining competition has the right to update the chain with a new block. In addition, the consensus protocol prescribes that the "longest" chain proposed by the network will be accepted as the trusted public record. A chain is considered to be the longest one if it has incorporated the most "work" by miners or, equivalently, has burnt the most resources. This ensures that there is agreement within the peers of the network about what constitutes the true history of all past transactions.

The second element is a confirmation lag. This lag helps prevent users from altering the history of transactions through double spending. After having transferred ownership of an asset, a user can attempt to convince other users to accept an alternative history in which the transaction has not been conducted. For example, in the case of Bitcoin, a dishonest user can try to revoke a payment in Bitcoin after he has received goods from a seller. To do so, he needs to create an alternative history of transactions by winning the mining competition against all honest miners. If such an attack succeeds, the dishonest user effectively steals the goods from the seller, as he gets the goods without paying the seller. The seller can protect himself against double spending by delaying the delivery of goods until multiple confirmations of the trade have been recorded in the blockchain. This is the case when several new blocks have been stacked onto the block that contains the original record of the transaction. Double spending would then require waiting for the delivery of the good and then replacing all these other blocks and the original block containing the transaction.

The final aspect is that validators need to be provided the proper incentive. Under a PoW protocol, the probability of winning the right to update a block is proportional to the fraction of computational power owned by a miner. Hence, sufficient overall mining activities help discourage dishonest behaviour. Since mining is costly, it has to be induced by offering rewards. These rewards can either be financed by seigniorage — issuing cryptocurrency or tokens — or by collecting transaction fees from traders. Importantly, increasing rewards will increase the effort and, hence, the computational investment by miners, making it harder to double spend.

The key innovation of blockchain technology is to put the users in the system in charge of guarding the system itself. Nakamoto (2008) formalized a solution to the problem of having to trust a third-party as the guardian of a payment system. Beyond the original Bitcoin proposal, it has become clear that his somewhat brilliant idea applies more broadly. Ironically, it arrived at about the same time as Leo Hurwicz delivered his Nobel lecture that posed the problem of "But Who Will Guard

the Guardians?" once again (see Hurwicz (2008)). Our attempt here is to go full circle by linking the idea of a blockchain back to an economic mechanism design problem.

In what follows, we first express the PoW protocol as a simple Cournot game of mining. We then formalize the double-spending problem and show that it can be summarized as a simple incentive compatibility constraint given the Cournot game of mining. We then briefly describe how this constraint manifests itself in two examples. The first example concerns a securities settlement system where both the asset transfer and the payment are recorded on a blockchain. The second one is the classic example of using a cryptocurrency for payments as intended in the original Bitcoin proposal. We close out our contribution by briefly discussing some further issues.

2 Modelling the Proof-of-Work Protocol

We first set up a simple game form[1] that captures the basic idea behind the PoW protocol where consensus on the blockchain is reached by a competition called mining. Time is continuous and there are M miners. The *protocol* specifies a computational problem and sets a difficulty D for the problem. It also specifies a reward R for the first miner to solve the problem and decrees that this miner is allowed to add a new block to the blockchain.

At the start of time, each miner invests a quantity q_i, $i = 1, \ldots, M$, into computing power to solve the computationally difficult problem. We denote the price per unit of computing power by α. The probability that miner i with computing power q_i solves the computational problem in a given time t is assumed to follow an exponential distribution with parameter q_i/D. Hence, a miner can solve the problem before t with probability

$$F(t) = 1 - e^{-\frac{q_i}{D}t}. \tag{1}$$

The expected time for a solution by miner i is then given by D/q_i.

The first solution (or proof-of-work (PoW)) among all M miners is then also an exponentially distributed random variable with parameter $\frac{1}{D}\sum_{i=1}^{M} q_i$. Given $\{q_i\}_{i=1}^{M}$, the expected time needed to complete the PoW is

$$\frac{D}{\sum_{i=1}^{M} q_i}, \tag{2}$$

[1] We use this terminology in the spirit of Hurwicz (2008) since the protocol specifies the strategy domain for the individual actors and the outcome function that maps strategies into outcomes.

with miner i having the probability

$$\rho_i \equiv \frac{q_i}{\sum_{i=1}^{M} q_i} \tag{3}$$

of being the first one to solve the problem. By adjusting the parameter D, the protocol can thus ensure that – on average – a solution is found in a particular time interval.

The PoW protocol essentially formalizes an exponential race where miners decide how much to invest into the race to win a reward R. This implements a simple Cournot game where each miner maximizes his payoff

$$\max_{q_i} \rho_i R - \alpha q_i, \tag{4}$$

taking as given the investments of all other miners $-i$. The symmetric Nash equilibrium to the Cournot game is given by

$$q_i = Q \equiv \frac{M-1}{\alpha M^2} R, \tag{5}$$

which leads to the following result.

Lemma 1. *The total mining cost is given by*

$$C \equiv \alpha Q M = \frac{M-1}{M} R$$

and the expected time to solve a block (i.e., block time) is

$$T \equiv \frac{\alpha M}{M-1} \frac{D}{R}.$$

This allows us to immediately derive some comparative statics for the PoW protocol. Of interest are the role of (i) computing costs, (ii) rewards, and (iii) total mining capacity. Most importantly, as the number of miners increases, the total mining cost converges to R. Competition dissipates all rents from mining, with miners earning zero expected profits when $M \to \infty$. These results are summarized below.

Proposition 2. *The block time D decreases when mining rewards R increase, when the number of miners M increases, or when the cost of computing power α falls.*

Total mining costs are unaffected by the cost of computing α, but increase proportionally with rewards R.

Total mining costs also increase with the number of miners M and mining profits converge to 0 as $M \to \infty$.

The mining game we have formalized takes place for each block and does not directly depend on block time (i.e., the frequency at which new blocks are added) and block size (i.e., the number of transactions that can be included in a single block). It does, however, crucially depend on the reward R for solving a block. This reward can be financed either through the issuance of tokens on the blockchain or through transaction fees posted by users for including their transactions into a block.

For tokens to serve as rewards, they need to have some real value. This value derives traditionally from their use as a payment instrument or, in other words, *cryptocurrency*. Users exchange their tokens against real goods. More recently, *initial coin offerings* have used tokens akin to shares in crowd investments where future tokens can be seen as additional shares being offered. Transaction fees arise when block size and block time are used to make settlement a scarce resource. Restricting block size and lengthening block time create *congestion* on the blockchain and, thus, exploit users' willingness to pay for fast settlement.

In general, what matters is the reward per block. The difficulty D controls the reward that is available over a fixed amount of time. Note that D cannot be arbitrarily short. In a distributed network, due to network latency, it takes time for data to get from one designated point to another. Hence, some time delay is necessary to communicate updates of the blockchain and to ensure that all miners and users work with the same information on the blockchain. A lower difficulty speeds up settlement, but reduces user willingness to post transaction fees, and thus the reward. Similarly, any reward from newly created tokens needs to be split across blocks. Keeping block rewards fixed over a time interval, a faster block time will lower the rewards available per block.

Nakamoto (2008) was first to introduce the idea of using a PoW protocol for achieving consensus on a blockchain with his Bitcoin proposal. The PoW problem to be solved is to find a particular output to the SHA256 algorithm. This algorithm takes an input of any bit size and hashes it to produce a random, but unique and not invertible 256-bit output. Bitcoin's protocol requires a miner to produce a hash with a certain number of leading zeros using the transaction data in the block, a summary of the previous block, the blockheader, plus an additional random number. The difficulty is given by how many leading zeros the hash has to have. It is adjusted every 2016 blocks so that the average time it takes miners to solve a block is about 10 minutes.

The reward that the Bitcoin protocol offers for miners comes from two sources. First, each block includes a special transaction creating a certain number of new bitcoins. In other words, winning a

block creates seigniorage for the winning miner. Second, users pledge transaction fees so that their transactions are included quickly into a block. The winner of a block also wins these transaction fees. As pointed out in Proposition 2, if the value of Bitcoin relative to the dollar costs of computing power increases, competition among miners will go up. The protocol then has to raise the difficulty for solving the problem to ensure that the target for a block time of 10 minutes is achieved on average.

3 Ruling Out Incentives to Cheat

3.1 Double-Spending Problem

In a *permissionless* blockchain, any user can act as a miner to validate and process transactions. As pointed out before, cryptography ensures that only private key holders can transfer assets recorded on the blockchain. Therefore, a dishonest user cannot simply steal assets without stealing someone else's private keys. However, a user can still remove transactions that have been initiated by himself or by other users. To do so, he needs to alter the blockchain and have all other users believe in the altered blockchain. This refers to what we call a *double-spending problem*.

We will first describe the general problem of a user double spending and then consider two specific examples of the problem. Consider a blockchain based on a PoW protocol with mining reward R. Suppose that double spending requires a user to win the competition N times and gives an *additional payoff* denoted by Δ. The payoff for a user trying to double spend is given by

$$\Pi(N, R) = \max_{\tilde{\mathbf{q}}} \mathcal{P}[\tilde{\mathbf{q}}; N, Q(R)](NR + \Delta) - c(\tilde{\mathbf{q}}). \tag{6}$$

Here, the user chooses a vector of computing power $\tilde{\mathbf{q}}$ for the N blocks to maximize the net expected return, where the probability of success $\mathcal{P}(\tilde{\mathbf{q}}; N, Q)$ increases with $\tilde{\mathbf{q}}$ and decreases with N and Q. When the user successfully wins the mining game $N + 1$ times, he gains Δ from altering the blockchain. But he also receives all the block rewards from winning the mining game $N + 1$ times. Finally, the user needs to incur a mining cost $c(\tilde{\mathbf{q}})$.

A user has an incentive to double spend whenever $\Pi(N, R) > 0$. Hence, in order to rule out users tampering with the blockchain, we require that

$$\Pi(N, R) \leq 0, \tag{7}$$

which we call a *no-double-spending constraint* (NoDS). This is akin to an incentive constraint where the design of the blockchain – in particular, rewards and the confirmation lag – plays a crucial role in ruling out incentives for users to engage in double spending.

3.2 Example 1: Blockchain for Securities Settlement

We first study an example in which a blockchain is used for settling asset trades (Chiu and Koeppl, 2018). In any security settlement system, it is important to avoid settlement failures where the seller of a security fails to deliver the security while receiving payment, or the buyer of a security fails to deliver payment while receiving the security. A delivery versus payment (DvP) mechanism typically ensures that the security and cash are exchanged simultaneously in order to avoid such a settlement failure. When both the cash and the asset are recorded on the same blockchain, DvP can be enforced by a self-enforcing, autonomous program often called a *smart contract*. In particular, the two legs involving the transfer of security and the cash payment are executed either in their entirety or not at all. In database systems, this is referred to as an *atomic transaction*, which is an indivisible and irreducible series of database operations such that either all or none of them occurs.

With settlement on a blockchain, when two counterparties agree to exchange a payment for a security, they jointly broadcast a transaction message about the terms of trade to the network so that the miners will validate the transaction and update the blockchain accordingly. Since the transfer of the asset and the payment are linked in an atomic transaction, it is infeasible for one side to undo the joint transfers unilaterally. However, any counterparty can eliminate the entire transaction by mining a block that changes ownership of one of the legs (cryptocurrency or security) so that the original transaction is invalid. If such double spending is successful before the original transaction has been included in the blockchain, the transaction has effectively never occurred.

The buyer in a security trade has an incentive to double spend if the agreed price is higher than the current value of the security, while a seller has an incentive to do so when the current value of the security is higher than the price received in the trade. For example, suppose the two counterparties originally have agreed to trade the security at a price p. After the arrival of new information, the buyer's valuation of the security becomes v_b while the seller's valuation becomes v_s. In this example, the buyer has an incentive to revoke the trade – and effectively default – if $p - v_b > 0$. The seller wants to cancel the trade when $v_s - p > 0$. Hence the maximum incentive for one of the

two sides to double spend is

$$V = \max\{p - v_b, v_s - p\}. \tag{8}$$

As pointed out, an investor can revoke a transaction by simply including a message into a block that changes the ownership of the security or the cryptocurrency used for payment. Hence, a dishonest investor needs to win the mining game against honest miners just once. The probability of a successful double spend is therefore

$$\mathcal{P}(\tilde{q}; N = 1, Q(R)) = \frac{\tilde{q}}{\tilde{q} + QM}, \tag{9}$$

where $\tilde{q}$ is the computing power invested by the dishonest investor who attempts to revoke the transaction. In general, a dishonest user might be subject to a higher mining cost than a regular miner. In addition, a dishonest investor may suffer reputational damage or a penalty if cheating is detected. We can capture this by assuming that the mining cost per block is given by

$$c(\tilde{q}) = \Gamma + \hat{\alpha}\tilde{q}, \tag{10}$$

with $\Gamma \geq 0$ and $\hat{\alpha} \geq 1$.

The dishonest user therefore solves

$$\Pi(N = 1, R) = \max_{\tilde{q}} \frac{\tilde{q}}{\tilde{q} + QM}(R + V) - \Gamma - \hat{\alpha}\tilde{q}. \tag{11}$$

For an interior solution we obtain

$$\tilde{q} = QM\left(\sqrt{\frac{V + R}{\hat{\alpha}QM}} - 1\right), \tag{12}$$

with the gain from double spending given by

$$\Pi(N = 1, R) = \frac{\sqrt{\frac{V+R}{\hat{\alpha}QM}} - 1}{\sqrt{\frac{V+R}{\hat{\alpha}QM}}}(V + R) - \Gamma - \hat{\alpha}QM\left(\sqrt{\frac{V + R}{\hat{\alpha}QM}} - 1\right). \tag{13}$$

Proposition 3. *Suppose the cost function for double spending is given by $c(\tilde{q}) = \Gamma + \hat{\alpha}\tilde{q}$ while the cost function for honest mining is $c(q) = q$.*

As $M \to \infty$, the NoDS constraint for users is given by

$$V \leq \Gamma + 2\sqrt{R\hat{\alpha}\Gamma} + R(\hat{\alpha} - 1).$$

The proposition implies directly that in order to discourage users from cheating, the fixed cost Γ or the marginal cost $\hat{\alpha}$ has to be sufficiently high. In particular, if double spending has no cost disadvantage over honest mining, one cannot rule out double spending in a securities settlement system based on a blockchain built on a PoW protocol.

3.3 Example 2: Blockchain for Cryptocurrency in Goods Transactions

We now consider a different example where a blockchain is used to record cryptocurrency transfers when purchasing real goods (Chiu and Koeppl, 2017). DvP in this context is not automatic anymore as the ownership of goods is not recorded digitally on the blockchain. Consider a spot trade where a buyer agrees to pay p units of cryptocurrency to a seller for a certain amount of goods. The buyer can cheat by mining a block where the transfer of cryptocurrency is not included, but instead the cryptocurrency is sent back to the buyer. If the attempt fails, the buyer pays the seller, but still gets the goods. If the attempt succeeds, the buyer gets the goods without paying the seller at all. Hence, the double-spending payoff for the buyer is the price of the goods, p, which effectively means he steals the goods.

Since there is no DvP, a seller can make double spending more difficult by introducing a confirmation lag. The goods are to be delivered only after the transaction has been confirmed sufficiently many times – say $N-1$ times – in the blockchain. This forces the buyer to win the mining game N times in a row, keeping it a secret from the rest of the miners each time he finds a solution in the first $N-1$ computational problem. Hence, we call a double-spending attempt *secret mining*. This way the seller is fooled to deliver the goods after $N-1$ confirmations, only to lose the payment in the following block.

Suppose now for simplicity that the cost of secret mining for a dishonest buyer is the same as that of a regular miner,

$$c(\tilde{q}) = \tilde{q}, \tag{14}$$

where we have normalized $\alpha = 1$. If there are no confirmation lags (i.e., $N = 0$), then Proposition 3 from Section 3.2 above immediately implies that buyers have no incentives to double spend if and only if $p < 0$. Hence, confirmation lags are necessary for preventing double spending.

To see the effect of a confirmation lag, suppose $N = 2$ so that the buyer needs to win the mining game twice in order to reclaim the payment p. If he succeeds, he earns $p + 2R$, while if he fails, he

earns 0. The dishonest buyer chooses his investment in computing power for the first and second blocks $(\tilde{q}_1, \tilde{q}_2)$ sequentially. The probability of a successful double-spending attempt is given by

$$\mathcal{P}(\tilde{q}_1, \tilde{q}_2; N = 2, Q) = \frac{\tilde{q}_1}{\tilde{q}_1 + QM} \frac{\tilde{q}_2}{\tilde{q}_2 + QM}, \tag{15}$$

as the buyer needs to win both blocks in order to revoke the payment.[2] We can solve the problem backward, starting from the second block. Conditional on having solved the first block, the optimal secret mining investment $\tilde{q}_2$ for the second block is a solution to the following problem:

$$\max_{\tilde{q}_2} \frac{\tilde{q}_2}{\tilde{q}_2 + QM}(p + 2R) - \tilde{q}_2. \tag{16}$$

Hence, the optimal investment is positive and given by[3]

$$\tilde{q}_2 = QM \left(\sqrt{\frac{p + 2R}{QM}} - 1 \right). \tag{17}$$

The expected payoff from secret mining is also positive and given by

$$\Pi_2 = QM \left(\sqrt{\frac{p + 2R}{QM}} - 1 \right)^2. \tag{18}$$

Given this solution, the optimal investment in secret mining for the first block solves

$$\max_{\tilde{q}_1} \frac{\tilde{q}_1}{\tilde{q}_1 + QM} \Pi_2 - \tilde{q}_1. \tag{19}$$

Hence, the optimal investment is given by

$$\tilde{q}_1 = \max \left\{ QM \left(\sqrt{\frac{p + 2R}{QM}} - 2 \right), 0 \right\}, \tag{20}$$

and the expected payoff from mining secretly is

$$\Pi(N = 2, R) = \max \left\{ QM \left(\sqrt{\frac{p + 2R}{QM}} - 2 \right)^2, 0 \right\}. \tag{21}$$

It follows immediately that $\tilde{q}_1 < \tilde{q}_2$ because the chance of successful double spending has gone up once the buyer has successfully mined the first block. The following proposition derives a constraint on the buyer to rule out double spending.

---

[2]This is only an approximation of the decision problem of double spending. See Section 4 for a discussion of the issue.

[3]Note also that the buyer – having started to mine secretly – has no incentive to announce that he has found a block. He would immediately get the block reward R but void the transaction, which is worth at least p. Hence, conditional on winning the first block with secret mining, the buyer has an incentive to keep on mining secretly, independent of $p > 0$. For details, see Chiu and Koeppl (2017).

Proposition 4. *Suppose there is one confirmation lag so that $N = 2$ and suppose the mining costs for double spending are $c(\tilde{q}) = \tilde{q}$.*

As $M \to \infty$, the NoDS constraint for users is given by

$$p < 2R.$$

Therefore, requiring confirmation in at least one block, which means that a double spending has to solve $N = 2$ blocks to be successful, the transaction is double-spending proof if the payment size is small relative to the mining rewards. Chiu and Koeppl (2017) show that, for any given N, the NoDS constraint is given by

$$p < RN(N-1). \tag{22}$$

Hence, larger transaction sizes require longer confirmation lags or higher rewards for mining to ensure that there are no incentives for buyers to double spend.

4 Discussion

4.1 The Costs of Cryptocurrencies

Keeping records on a blockchain is not a free lunch. It is necessary to offer rewards to rule out double spending, which directly or indirectly increases the cost of maintaining a distributed ledger. In the case of a cryptocurrency, rewards can be offered by seigniorage. Such seigniorage causes inflation, which levies indirect costs in the form of an inflation tax on users. However, there are also direct costs that arise from investment into computational power (mainly energy) that uses up most of the revenue from seigniorage. A traditional currency does not waste seigniorage, but raises revenue for the issuer. In the case of a modern central bank, this generates profits above operational costs that can be used to offset other distortionary taxes used by the government. A quantitative assessment has shown that a low-inflation currency regime dominates any cryptocurrency (see Chiu and Koeppl (2017)).

Still, cryptocurrencies can exploit a trade-off between settlement speed and rewards to deter double spending. As shown in Section 3.3, for any given transaction, increasing the confirmation lag reduces the reward necessary for a tamper-proof blockchain. Nevertheless, increasing confirmation lags has a time cost as the delivery of goods is delayed.

This points to settling transactions in cryptocurrencies being a public good. Settling one transaction does not preclude settling more transactions at any given time. Interestingly, double spending is driven by transactions that have the largest incentives to do so. Hence, all other transactions with lower incentives can free-ride once double spending has been ruled out. In contrast, however, a single transaction with very large incentives to double spend requires very large rewards for mining blocks. These costs are then indirectly borne by all other users, which can make using a cryptocurrency unnecessarily expensive. This implies that a cryptocurrency works best for a fairly homogeneous group of transactions with small incentives to double spend, such as retail payments (see again Chiu and Koeppl (2017)).[4]

4.2 Double Spending as a Poisson Race

For simplicity, we have modelled secret mining as an exponential race for each update against a group of honest miners. In order to double spend, a user had to win the race N times in a row, but keep his result secret. This is not entirely accurate when looking at actual PoW protocols employed for blockchain technology. These users can catch up and only need to generate at least N blocks faster than all the other miners. Hence, secret mining is really a Poisson race against a fringe of honest miners that play a sequence of simple exponential races to find one block in each race.

This is related to the so-called "51% attack" problem. If a miner controls more than half the computational power among all miners, confirmation lags, in theory, lose their power in controlling double-spending incentives. The dishonest miner creates an arrival rate that is larger than those of the other honest miners combined. This implies that he is certain to eventually outrun other miners in generating a longer chain and, thus, can always cheat by double spending (see, for example, Rosenfeld (2014)). However, from an economic point of view, this requires that a dishonest miner has deep pockets and is risk neutral. These assumptions tend to be unrealistic and, in practice, users have little economic incentives to launch such an attack, especially when the computational investment by other miners is large.

This is reflected in our approach where larger confirmation lags reduce the probability of successful double spending. More generally, rather than ruling out double spending altogether, it could be

[4]In other applications, like securities settlement systems, it may be necessary to counteract the public good character of settling on a blockchain. One has to create some congestion so that users have an incentive to post transaction fees. Settlement on a blockchain then becomes a club good.

sufficient to ensure that double spending only occurs with a sufficiently small probability. Interestingly, there could then even be competition for double spending where there are multiple dishonest users. If coordination of such behaviour is difficult, then double spending from the perspective of an individual transaction is small.

4.3 Other Consensus Protocols

We have focussed exclusively on PoW protocols to achieve consensus. Many other protocols have been discussed that try to save on the costs associated with running a blockchain. Protocols based on *proof-of-stake* (PoS) allocate the right to update the blockchain randomly across users. The chance of any user to win the right is linked to his stake in the system, for example, the number of units of cryptocurrency the user owns. However, these alternative systems usually do not possess a key feature of PoW: one needs to spend a large amount of resources to be successful in cheating, and being unsuccessful means that one has incurred a large, irretrievable sunk cost.

Another alternative is a type of voting arrangement where a majority or super-majority of users are needed to agree on a new block. The classic protocol in this area is *Practical Byzantine Fault Tolerance* (PBFT), where for an update, two-thirds of the users in a network need to agree that two-thirds of the users have agreed on a new block.[5] However, any blockchain with too many nodes cannot implement such a protocol as it introduces too much latency due to extreme communication requirements. Consequently, such protocols have been explored mainly in "closed" or "permissioned" blockchains, where a small group of known validators are charged with updating the blockchain.

References

Chiu, J and Koeppl, T (2017) The Economics of Cryptocurrency – Bitcoin and Beyond. Queen's University, Working Paper 1389

Chiu, J and Koeppl, T (2018) Blockchain-based Settlement for Asset Trading. Queen's University, Working Paper 1397

[5]The consensus protocol is built to tolerate failures of up to 33% of nodes in a distributed system. This is the theoretical limit of failures such a system can sustain when having some form of syncronization (see Lamport et al. (1982)).

Hurwicz, L (2008) But Who Will Guard the Guardians? American Economic Review, 98:577–585

Lamport, L, Shostak, R and Pease, M (1982) The Byzantine Generals Problem. ACM Transactions on Programming Languages and Systems, 4:382–401

Nakamoto, S (2008) Bitcoin: A Peer-to-Peer Electronic Cash System. White Paper

Rosenfeld, M (2014) Analysis of Hashrate-based Double Spending. arXiv:1402.2009